A Tale of Two Metrics in Network Delay Optimization

Qingyu Liu*†, Lei Deng‡†, Haibo Zeng*, Minghua Chen†
*Department of Electrical and Computer Engineering, Virginia Tech, USA
†Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong
‡School of Electrical Engineering & Intelligentization, Dongguan University of Technology, China

***Abstract*—We consider the scenario where a source streams a flow at fixed rate to a receiver across a network, possibly using multiple paths. Transmission over a link incurs a delay modeled as a non-negative, non-decreasing and differentiable function of the link aggregated transmission rate. This setting models various practical network communication scenarios. We study network delay optimization concerning two popular metrics, namely maximum delay and average delay experienced by the flow. A well-known pessimistic result says that a flow cannot simultaneously achieve optimal maximum delay and optimal average delay, or even within constant-ratio gaps to the two optimums. In this paper, we pose an optimistic note on the fundamental compatibility of the two delay metrics. Specifically, we design two polynomial-time solutions to deliver $(1-\epsilon)$ fraction of the flow with maximum delay and average delay simultaneously within $1/\epsilon$ to the optimums for any $\epsilon \in (0,1)$. Hence, the two delay metrics are "largely" compatible. The ratio $1/\epsilon$ is independent to the network size and link delay function, and we show that it is tight or near-tight. Simulations based on real-world continent-scale network topology verify our theoretical findings. Note that the proposed delay gap $1/\epsilon$, upon sacrificing ϵ fraction of the flow rate, is guaranteed even under the worst theoretical case setting. Our simulation results show that the empirical delay gaps observed under practical settings can be much smaller than $1/\epsilon$. Our results are of particular interest to delay-centric networking applications that can tolerate a small fraction of traffic loss, including cloud video conferencing that recently attracts substantial attention.**

I. Introduction

A. Background and Motivation

We consider the scenario where a source streams a flow at fixed rate to a receiver across a network, possibly using multiple paths. Transmission over a link incurs a delay modeled as a *non-negative, non-decreasing, and differentiable* function of the link aggregated transmission rate. We study three fundamental network delay optimization problems, concerning two popular metrics, namely maximum delay and average delay. The three problems are (i) minimizing the maximum delay, denoted as problem MM, (ii) minimizing the average delay, denoted as problem SO, and (iii) finding the Nash equilibrium, denoted as problem NE. Our focus is on understanding the fundamentals of the two delay metrics. Specifically, we study quality of the solutions to the three problems, in terms of simultaneously attaining optimal or close-to-optimal maximum delay and average delay performances.

Our study is motivated by recent skyrocketing interests on supporting delay-centric traffics in data center networks [1], [2] and real-time communications such as cloud video conferencing [3], [4]. For example, it is reported that 51 million users per month attend WebEx meetings [5], 3 billion minutes of calls per day use Skype [6], and 75% of high-growth innovators use video collaboration [7]. As recommended by International Telecommunication Union (ITU) [8], for highly interactive tasks including video conferencing, it is desirable to keep the cross-network one-way delay as low as possible. A delay less than 150ms can provide a transparent interactivity while delays above 400ms are unacceptable [8].

The communication network delay is composed of the processing delay, packet queuing delay, and propagation delay. In many cases, queuing delay is the dominant factor, which increases with the packet flow rate. According to queuing theory, the link delay can be estimated by the following function (for M/M/1 queue together with a FIFO server) [9]

$$\mathcal{D}(x) = \begin{cases} \frac{1}{c-x} & \text{if } c > x, \\ +\infty & \text{otherwise,} \end{cases} \tag{1}$$

where c is the capacity and x is the assigned traffic. Function (1) does not allow the overhead case where assigned rate exceeds the capacity. Better but more complex delay formulas have been proposed to account for the overhead case and propagation delay [10]. In this paper, we design traffic routing strategies for arbitrary link delay functions that satisfy some modest assumptions (*non-negative, non-decreasing, and differentiable*). These assumptions are satisfied in almost all settings including the function in (1) and those in [10].

B. Network Delay Optimization

The two delay metrics, i.e., maximum delay and average delay, are optimized by the three fundamental network optimization problems, i.e., MM, SO and NE, in different ways and are of strong practical relevance in delay-centric networking applications. For example, for traffic routing in cloud video conferencing, the optimum to MM provides a good assignment for conferencing with multiple sessions since the maximum end-to-end delay is minimized. Hence all sessions can experience a delay within an optimal upper bound. SO minimizes the average end-to-end delay and gives an optimum to efficiently utilize the network resources from the view of the cloud. The optimum to NE is the assignment to which delay-aware distributed rate control protocols converge for large networks without centrally administered servers. It provides the best fair solution where conferencing sessions with the same source and destination will experience the same delay.

In terms of computational complexity (see Tab. I), [11] states that MM is NP-hard, NE can be formulated as a convex program and thus solved in polynomial time, and SO is also a convex program if the link delay function is convex. Thus,

TABLE I: Problem complexity [11].

MM	NP-hard
NE	Convex program
SO	Convex program if all link delay functions are convex

TABLE II: Existing results from [11], [12].

	Maximum delay bound	Average delay bound
MM	1	$\sigma(\mathcal{L})$ (tight)
NE	$\sigma(\mathcal{L})$ (tight)	$\sigma(\mathcal{L})$ (tight)
SO	$\gamma(\mathcal{L})$ (tight)	1

NE and SO are easy to solve, but it is impossible to obtain the optimum to MM in polynomial time unless P = NP.

In delay-centric applications, an ideal traffic routing approach should optimize both the maximum delay and the average delay, benefiting each user as well as the overall network simultaneously. Specifically, to ensure that all users have a satisfactory experience, maximum end-to-end delay shall be minimized or at least bounded above by the tolerant value (e.g., 400ms in video conferencing). Also, it is desirable to provide close-to-optimal average delay to ensure an efficient utilization of network resources.

Since only the NP-hard problem MM optimizes the maximum delay, a key question for delay-centric routing is to design efficient approximation algorithms to approximately solve MM and simultaneously provide a close-to-optimal average delay performance. Because SO and NE are easy to solve, a natural idea is to see whether their solutions can provide certain maximum delay performance and average delay performance guarantee. Along this line, next we summarize existing results (Section I-C) and show our contributions (Section I-D).

C. Existing Results

For NE, Roughgarden [13] shows that the Nash equilibrium admits a network-dependent maximum delay bound of $(|V|-1)$ where $|V|$ is the number of nodes in the network. Roughgarden and Tardos [14] further prove that the Nash equilibrium provides an average delay bound of $4/3$ for linear link delay functions. Correa *et al.* in [11], [12] study arbitrary link delay functions and their results are shown in Tab. II in terms of both maximum delay and average delay bounds. In Tab. II, $\mathcal{L}$ is the set of all link delay functions, and both $\gamma(\mathcal{L})$ and $\sigma(\mathcal{L})$ depend on $\mathcal{L}$. Although their bounds $\gamma(\mathcal{L})$ and $\sigma(\mathcal{L})$ have been proved to be tight, we can see that they rely on the link delay model. Even worse, they can be arbitrarily large for certain delay functions [15].

Overall, we observe pessimistically that a flow cannot simultaneously achieve optimal maximum delay and average delay, or even within constant-ratio gaps to the two optimums, for general network topologies and arbitrary link delay models.

D. Our Contributions

In this paper, we pose an optimistic note on the fundamental compatibility of the two delay metrics. Specifically, we propose a new approach to *achieve constant maximum delay bound and constant average delay bound simultaneously* by *sacrificing the flow rate within a controllable level.* We summarize our results in Table III. Our approach is to either delete a controllable portion of rates from the flow solutions (Thm. 3 and 4 for DF, i.e., Deleting Flow from SO), or directly solve the flow problems with a controllably smaller rate (Thm. 5 and 6 for NE, Thm. 7 and 8 for MM).

In this way, for each of the three problems including SO, NE and MM, we can obtain a solution which has theoretically guaranteed maximum delay and average delay that are simultaneously close to the two optimums, only in the cost of losing a controllable portion of the rate. For arbitrarily link delay functions and general network topologies, all proposed delay bounds are constants which are at least near-tight, an optimistic and different result than existing studies (e.g. [11], [12]) which cannot bound the two delay metrics simultaneously within constant gaps to the two optimums without sacrificing flow rate. We believe that these results are of particular interest to delay-centric networking applications that can tolerate a small fraction of traffic loss, including cloud video conferencing. Simulations based on real-world continent-scale network topology verify our theoretical findings. Besides, our simulation results show that the empirical delay gaps observed under practical settings are much smaller than $1/\epsilon$, which is the theoretical gap guaranteed even in the worst case upon sacrificing ϵ fraction of the flow rate, calling for possible future work to improve our proposed delay gaps with more practical parameters involved when flow rate can be sacrificed.

As theoretical byproducts of our proposed delay bounds, we derive two polynomial-time bi-criteria approximation algorithms to solve the NP-hard problem MM. One is to solve the problem NE directly with a controllably smaller rate requirement, which has polynomial time complexity regardless of the convexity of the delay functions as long as they are non-negative, non-decreasing and differentiable. The other is our Algorithm 1 to delete a controllable portion of rate from the solution to SO, which has polynomial time complexity under an extra mild assumption that all link delay functions are convex. Both resulting flows have a constant and near-tight bi-criteria approximation ratio $(1-\epsilon, 1/\epsilon)$. To our best knowledge, this is the first time to design approximation algorithms to solve MM with constant approximation ratios independent of input networks and link delay functions.

The rest of the paper is organized as follows. Sec. II gives our system model and defines the three delay-centric network flow problems. Sec. III describes the algorithm to obtain the flow solution for DF and proves non-decreasing properties for its average and total delay. Sec. IV details the proof of the results in Table III. Sec. V presents our experiments, followed by the conclusion in Sec. VI. Finally in Appendix, we present lemmas and associated proofs which will be used heavily.

II. System Model and Problem Formulation

A. System Model

In this paper we model the input network as a directed graph $G \triangleq (V,E)$. Data transmission over a link $e \in E$ incurs a delay modeled by a function $\mathcal{D}_e(x_e)$ that is *non-negative, non-decreasing, and differentiable* w.r.t. the aggregated link rate x_e. We study the single-unicast networking scenario where a

TABLE III: Our results: Bi-criteria delay bounds.

	Average delay bound compared to $f_{\mathsf{SO}}(R)$			Maximum delay bound compared to $f_{\mathsf{MM}}(R)$		
	Ratio	Proof	Tightness Example	Ratio	Proof	Tightness Example
$f_{\mathsf{SO}}[(1-\epsilon)R]$	$(1-\epsilon, 1)$	Thm. 1	Fig. 1	$(1-\epsilon, \gamma(\mathcal{L}))$	Thm. 2	Unknown
$f_{\mathsf{DF}}[(1-\epsilon)R]$	$(1-\epsilon, 1)$	Thm. 3	Fig. 1	$(1-\epsilon, 1/\epsilon)$	Thm. 4	Fig. 2 *
$f_{\mathsf{NE}}[(1-\epsilon)R]$	$(1-\epsilon, 1/\epsilon)$	Thm. 5	Fig. 3	$(1-\epsilon, 1/\epsilon)$	Thm. 6	Fig. 2 *
$f_{\mathsf{MM}}[(1-\epsilon)R]$	$(1-\epsilon, 1/\epsilon)$	Thm. 7	Fig. 3	$(1-\epsilon, 1)$	Thm. 8	Fig. 1

Note. *: the figure is used to prove near-tightness instead of tightness.

source node $s \in V$ streams a flow of rate $R > 0$ to a receiver $t \in V \backslash \{s\}$, possibly using multiple paths in G.

We define P as the set of all paths from s to t. A flow solution f is defined as the assigned rates over P, i.e., $f \triangleq \{x^p : x^p \geq 0, p \in P\}$, where x^p is the rate on path p. The total rate sent from s to t under flow f is defined as

$$|f| \triangleq \sum_{p \in P} x^p.$$

The aggregated assigned rate on link e under flow f is

$$x_e \triangleq \sum_{p \in P: e \in p} x^p,$$

and the path-p delay under flow f is

$$d^p(f) \triangleq \sum_{e \in p} \mathcal{D}_e(x_e),$$

which is the sum of delays for all links belonging to p.

B. Maximum Delay and Average Delay

The *maximum delay* of the flow f is the maximum path delay among all paths that have a positive rate[1], defined as

$$\mathcal{M}(f) \triangleq \max_{p \in P: x^p > 0} d^p(f). \tag{2}$$

The *total delay* of flow f is the sum of delays experienced by all flow units, defined as

$$\mathcal{T}(f) \triangleq \sum_{p \in P} d^p(f) \cdot x^p = \sum_{e \in E} \mathcal{D}_e(x_e) \cdot x_e. \tag{3}$$

The *average delay* of flow f is the ratio between the total delay and the total flow rate, defined as

$$\mathcal{A}(f) \triangleq \frac{\mathcal{T}(f)}{|f|}. \tag{4}$$

C. Three Network Delay Optimization Problems

MM: the Minimal Maximum delay problem aims to minimize the maximum delay subject to a flow rate constraint

$$\mathsf{MM}: \quad \min_f \mathcal{M}(f) \quad \text{s.t.} \quad |f| = R.$$

We define $f_{\mathsf{MM}}(R)$ as an optimal flow solution to problem MM under rate requirement R. For convenience, $f_{\mathsf{MM}}(R)$ is called a *min-max flow*. We define $\mathcal{M}^*(R)$ as the optimal maximum delay that can be achieved for any feasible flow with rate R. Clearly, it is $\mathcal{M}^*(R) = \mathcal{M}(f_{\mathsf{MM}}(R))$.

[1]If a path has a positive rate, we call it *a flow-carrying path.*

SO: the System Optimization (minimal average delay) problem is to minimize the average delay (or equivalently the total delay) subject to flow rate constraint. It is formulated as

$$\mathsf{SO}: \quad \min_f \mathcal{A}(f) \quad \text{s.t.} \quad |f| = R.$$

We use $f_{\mathsf{SO}}(R)$ to denote an optimal flow solution to SO under rate requirement R. The minimal average delay that can be achieved for flows with rate R is denoted as $\mathcal{A}^*(R)$. To facilitate later analysis, we also define the minimal total delay as $\mathcal{T}^*(R)$. Clearly we have $\mathcal{A}^*(R) = \mathcal{A}(f_{\mathsf{SO}}(R))$, and $\mathcal{T}^*(R) = R \cdot \mathcal{A}^*(R) = \mathcal{T}(f_{\mathsf{SO}}(R))$. We also call $f_{\mathsf{SO}}(R)$ a *system-optimal flow*.

It is well known that SO can be described as an optimization problem using an edge-based flow formulation [16]. When all edge delay functions are convex, the edge-based formulation of SO is a convex program with polynomial size, and thus is easy to solve [14]. The convex program solution then can be converted to a path-based flow by a flow decomposition algorithm in polynomial time [17]. Note that the flow decomposition algorithm could output multiple path-based flows, all of which have the same average delay but may have different maximum delays. In this paper, we regard $f_{\mathsf{SO}}(R)$ as an *arbitrary* path-based flow after flow decomposition.

NE: to find a Nash Equilibrium flow subject to flow rate constraint. A Nash equilibrium flow is defined as follows.

Definition 1: A flow f is a *Nash equilibrium flow* (or in short a *Nash flow*) if for any pair of paths $p_1, p_2 \in P$ with $x^{p_1} > 0$, $d^{p_1}(f) \leq d^{p_2}(f)$.

The Nash flow is the quickest fair flow [11], where no flow unit has any incentive to deviate from its own path because there does not exist a path with a smaller delay.

Problem NE can be written as

$$\mathsf{NE}: \quad \text{find a Nash flow } f \text{ such that } |f| = R.$$

As proved in [14, Lemma 2.6], for all link delay functions that are non-negative, non-decreasing, and differentiable, there exists a Nash flow and all Nash flows have the same total/average/maximum delay. We use $f_{\mathsf{NE}}(R)$ to denote an arbitrary Nash flow under rate requirement R.

These three fundamental problems all aim to optimize network delay performance. Still, following Tab. II and the discussion in Sec. I-C, pessimistically none of them can simultaneously achieve optimal maximum delay and average delay, not even within constant-ratio gaps to the two optimums for general network topology and arbitrary link delay function. However, in this paper we pose an optimistic note on the two delay metrics that they are "largely" compatible and can be optimized to simultaneously guarantee within $1/\epsilon$ to the optimums, only in the cost of sacrificing ϵ fraction of the flow rate. Next we define approaches for each of the three problems to sacrifice rate such that both maximum delay and average delay can be close to optimums as proved later in Sec. IV.

D. Bi-Criteria Bounds: Bounded Delay with Sacrificed Rate

Definition 2: A flow f_1 is a (μ, ρ)-approximate solution to another flow f_2 in terms of maximum delay, or equivalently the bi-criteria maximum delay bound is (μ, ρ) for f_1 compared with f_2, if

$$|f_1| \geq \mu |f_2|, \tag{5}$$

$$\mathcal{M}(f_1) \leq \rho \mathcal{M}(f_2). \tag{6}$$

Bi-criteria average delay bound is defined in a similar manner.

In the definition, when $\mu < 1$, f_1 only sends a fraction of the rate of flow f_2. For any network topology and arbitrary delay function, in order to obtain optimization solutions with both maximum delay and average delay guaranteed to be close to optimums, we consider the possibility of sacrificing the rate requirement. More concretely, given any $\epsilon \in (0, 1)$ and the rate requirement R, we first are interested in the following three problems that directly solve the corresponding optimization problem with a reduced rate requirement $(1-\epsilon)R$:

- $f_{\mathsf{MM}}[(1-\epsilon)R]$: an optimal solution to problem MM with a rate requirement $(1-\epsilon)R$;
- $f_{\mathsf{SO}}[(1-\epsilon)R]$: an optimal solution to problem SO with a rate requirement $(1-\epsilon)R$;
- $f_{\mathsf{NE}}[(1-\epsilon)R]$: an optimal solution to problem NE with a rate requirement $(1-\epsilon)R$.

However, as in Tab. III, we are only able to guarantee the tightness (or near-tightness) and constantness for the bounds of $f_{\mathsf{MM}}[(1-\epsilon)R]$ and $f_{\mathsf{NE}}[(1-\epsilon)R]$, but not those of $f_{\mathsf{SO}}[(1-\epsilon)R]$. This motivates us in Sec. III to further design a novel flow sacrificing algorithm (Algorithm 1) to get a flow from problem SO with constant bi-criteria bounds:

- $f_{\mathsf{DF}}[(1-\epsilon)R]$: the solution by Deleting ϵR Flow rate in polynomial time from $f_{\mathsf{SO}}(R)$ using Algorithm 1.

As opposed to $f_{\mathsf{SO}}[(1-\epsilon)R]$, we are able to demonstrate that $f_{\mathsf{DF}}[(1-\epsilon)R]$ leads to tight or near-tight constant maximum delay and average delay bounds. Note that $f_{\mathsf{DF}}[(1-\epsilon)R]$ is *not* the system-optimal flow under rate $(1-\epsilon)R$.

III. A Novel Flow-Sacrificing Algorithm to SO

As motivated in Sec. II, in this section we propose a novel algorithm to reduce ϵR rate from $f_{\mathsf{SO}}(R)$, such that near-tight constant maximum delay and average delay bounds can be guaranteed as proved later in Sec. IV.

A. Procedure to Find $f_{\mathsf{DF}}[(1-\epsilon)R]$ in Polynomial Time

In Algorithm 1, first, we obtain a path-based system-optimal flow $f_{\mathsf{SO}}(R)$ given rate requirement R (Line 4). Then, the algorithm deletes ϵR rate iteratively from $f_{\mathsf{SO}}(R)$ (Lines 6–15). In each iteration, we find the slowest flow-carrying path p_l, i.e., the path that has the maximum path delay among all paths with a positive flow rate, and then delete the right amount

Algorithm 1 Finding $f_{\mathsf{DF}}[(1-\epsilon)R]$

```
1:  input: $G = (V, E)$, $R$, $s$, $t$, $\epsilon \in (0, 1)$
2:  output: $f_{\mathsf{DF}}[(1-\epsilon)R]$
3:  procedure
4:      $f_{\mathsf{SO}}(R)$ = System-Optimal-Flow($G, R, s, t$)
5:      $x^{\text{delete}} = \epsilon R$     //the rate to be deleted
6:      while $x^{\text{delete}} > 0$ do
7:          Find the slowest flow-carrying path $p_l$
8:          if $x^{p_l} > x^{\text{delete}}$ then
9:              $x^{p_l} = x^{p_l} - x^{\text{delete}}$
10:             $x^{\text{delete}} = 0$
11:         else
12:             $x^{\text{delete}} = x^{\text{delete}} - x^{p_l}$
13:             $x^{p_l} = 0$
14:         end if
15:     end while
16:     $f_{\mathsf{DF}}[(1-\epsilon)R]$ is defined by the remaining flow
17:     return $f_{\mathsf{DF}}[(1-\epsilon)R]$
18: end procedure
```

of rate from it. The iteration terminates when ϵR rate is deleted in total, and the remaining flow is $f_{\mathsf{DF}}[(1-\epsilon)R]$.

With the assumption that all link delay functions are convex, we can get a path-based system-optimal flow in polynomial time (Line 4) as discussed in Sec. II: after solving the convex program formulation of SO, we do a flow decomposition on its optimum. Note that the flow decomposition outputs at most $|E|$ paths and thus $f_{\mathsf{SO}}(R)$ contains at most $|E|$ flow-carrying paths [17]. Hence, the while loop (Lines 6–15) in Algorithm 1 terminates in at most $|E|$ iterations. In each iteration, we need to recalculate all flow-carrying path delays and find the path with largest delay, which takes $O(|E|)$ time. Thus, the while loop takes polynomial time. Overall, Algorithm 1 can output the solution $f_{\mathsf{DF}}[(1-\epsilon)R]$ in polynomial time in the network size $|V|$ and $|E|$, and its complexity is independent from ϵ.

B. Upper Bounded Average and Total Delay for $f_{\mathsf{DF}}[(1-\epsilon)R]$

In this subsection for the newly proposed $f_{\mathsf{DF}}[(1-\epsilon)R]$, we present the results comparing its average/total delay to the optimal average/total delay for flows with the full rate R.

Lemma 1: Following Algorithm 1, we have:

$$\mathcal{A}[f_{\mathsf{DF}}((1-\epsilon)R)] \leq \mathcal{A}^*(R), \tag{7}$$

$$\mathcal{T}[f_{\mathsf{DF}}((1-\epsilon)R)] + \epsilon R \cdot \mathcal{M}[f_{\mathsf{DF}}((1-\epsilon)R)] \leq \mathcal{T}^*(R). \tag{8}$$

Proof: See Appendix B. ■

Lem. 1 directly implies that the average delay of $f_{\mathsf{DF}}[(1-\epsilon)R]$ is guaranteed to be no worse than the optimal average delay under the full rate, leading to a bi-criteria average delay bound of $(1-\epsilon, 1)$ for $f_{\mathsf{DF}}[(1-\epsilon)R]$ (Thm. 3). In the next section we will further prove a near-tight bi-criteria maximum delay bound of $(1-\epsilon, 1/\epsilon)$ for $f_{\mathsf{DF}}[(1-\epsilon)R]$ (Thm. 4). Therefore, the polynomial-time Algorithm 1 provides a desirable traffic assignment strategy for delay-centric applications, in that the maximum delay performance is close to optimum (within a constant-ratio gap of $1/\epsilon$) while the average delay is optimal, only in a cost of losing ϵ portion of the traffic rate.

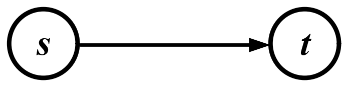


Fig. 1: A simple graph with one link and two nodes. It is used to show tightness in Thm. 1, Thm. 3, and Thm. 8 where the link delay function is a constant $\mathcal{D}(x) \equiv D$.

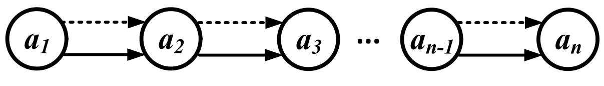


Fig. 2: A graph with n nodes and $2(n-1)$ links. All upper dashed links have a constant delay 1, while all lower solid links have a delay function $\mathcal{D}(x)$ defined in (15). It is used to prove the near-tightness in Thm. 4 and Thm. 6.

IV. Main Results: Bi-Criteria Delay Bounds

We study the two delay metrics, including the maximum delay and the average delay, when we sacrifice any $\epsilon \in (0,1)$ portion of flow rate for each of the three network optimization problems, leading to several tight or near-tight constant bi-criteria delay bounds independent of network topology and link delay function. The results are summarized in Table III.

Note that in the proof to any theorems of bi-criteria delay bounds in this section, the correctness that corresponding flow can support at least $(1-\epsilon)$ fraction of rate R is skipped, since it holds straightforwardly according to the definition of $f_{\mathsf{SO}}[(1-\epsilon)R]$, $f_{\mathsf{DF}}[(1-\epsilon)R]$, $f_{\mathsf{NE}}[(1-\epsilon)R]$ and $f_{\mathsf{MM}}[(1-\epsilon)R]$ presented in Sec. 2. Thus we only focus on proving the correctness of bounded delay and the tight or near-tight analysis.

A. Comparing $f_{\mathsf{SO}}[(1-\epsilon)R]$ with Optimum

For the system-optimal flow with a rate $(1-\epsilon)R$, we prove its average delay must be no worse than the optimum with full rate R in Thm. 1, but its maximum delay is bounded above by a delay-function-dependent ratio to the optimum in Thm. 2.

Theorem 1: Compared with $f_{\mathsf{SO}}(R)$, $f_{\mathsf{SO}}[(1-\epsilon)R]$ has a bi-criteria average delay bound $(1-\epsilon, 1)$, which is tight.

Proof: Since $f_{\mathsf{SO}}[(1-\epsilon)R]$ is the optimal solution to SO with a rate requirement $(1-\epsilon)R$ and $f_{\mathsf{DF}}[(1-\epsilon)R]$ is a feasible solution to SO with a rate requirement $(1-\epsilon)R$, we have

$$\mathcal{A}(f_{\mathsf{SO}}[(1-\epsilon)R]) \leq \mathcal{A}(f_{\mathsf{DF}}[(1-\epsilon)R]). \tag{9}$$

In addition, according to (7) in Lem. 1, we get

$$\mathcal{A}(f_{\mathsf{DF}}[(1-\epsilon)R]) \leq \mathcal{A}^*(R) = \mathcal{A}(f_{\mathsf{SO}}(R)). \tag{10}$$

Thus, (9) and (10) lead to

$$\mathcal{A}(f_{\mathsf{SO}}[(1-\epsilon)R]) \leq \mathcal{A}(f_{\mathsf{SO}}(R)). \tag{11}$$

Such a bound is tight for the network in Fig. 1, where the link delay function is a constant $\mathcal{D}(x) \equiv D$. For both $f_{\mathsf{SO}}[(1-\epsilon)R]$ and $f_{\mathsf{SO}}(R)$, the average delay is D. ■

Similar to the optimal maximum delay (Lem. 2) and the optimal total delay (Lem. 3), Thm. 1 says that the optimal average delay is non-decreasing w.r.t. the rate R.

Theorem 2: Compared with $f_{\mathsf{MM}}(R)$, a bi-criteria maximum delay bound for $f_{\mathsf{SO}}[(1-\epsilon)R]$ is $(1-\epsilon, \gamma(\mathcal{L}))$, where $\gamma(\mathcal{L})$ is the minimal number that meets: ($\hat{\mathcal{D}}(x)$ is defined in Lem. 6)

$$\hat{\mathcal{D}}(x) \leq \gamma(\mathcal{L}) \cdot \mathcal{D}(x), \forall \mathcal{D}(x) \in \mathcal{L}, \forall x \in [0,R] \tag{12}$$

Proof: The maximum delay bound $\gamma(\mathcal{L})$ can be proved directly from the maximum delay gap in Tab. II comparing SO to MM under the same rate and Lem. 2. ■

The delay-function-dependent maximum delay bound $\gamma(\mathcal{L})$ can be fairly large or even infinite in theory as explained in our report [15]. Although $\gamma(\mathcal{L})$ has been proved to be tight in [11] without sacrificing flow rate, whether it is tight remains unknown in our setting where we accept the cost of losing $\epsilon \in (0,1)$ portion of flow rate in order to obtain both maximum delay and average delay that are close-to-optimal.

Overall, to solve problem SO with a reduced rate does not necessarily result in an improved constant-bounded maximum delay gap for any network and arbitrary delay function. When applied to cloud video conferencing, solving SO between a pair of cloud relays with a reduced rate could still make users experiencing unacceptably large delay even when they are willing to loss some traffic.

B. Comparing $f_{\mathsf{DF}}[(1-\epsilon)R]$ with Optimum

Drawbacks from $f_{\mathsf{SO}}[(1-\epsilon)R]$ motivate us to design a new approach (Algorithm 1) to sacrifice the rate from the system-optimal flow, leading to the solution $f_{\mathsf{DF}}[(1-\epsilon)R]$. In this subsection, for $f_{\mathsf{DF}}[(1-\epsilon)R]$, we show its average delay is no worse than the optimum under full rate in Thm. 3, and furthermore its maximum delay is close-to optimal within a near-tight constant-ratio gap in Thm. 4.

Theorem 3: Compared with $f_{\mathsf{SO}}(R)$, a bi-criteria average delay bound for $f_{\mathsf{DF}}[(1-\epsilon)R]$ is $(1-\epsilon, 1)$, which is tight.

Proof: The correctness of the bound has been proved in Lem. 1. The tightness is proved similarly as in Thm. 1. ■

Theorem 4: Compared with $f_{\mathsf{MM}}(R)$, a bi-criteria maximum delay bound for $f_{\mathsf{DF}}[(1-\epsilon)R]$ is $(1-\epsilon, 1/\epsilon)$.

The bound $(1-\epsilon, 1/\epsilon)$ is near tight in the sense that for any $\epsilon \in (0,1)$, there exists a problem instance where the bi-criteria maximum delay bound is $(1-\epsilon, \lceil 1/\epsilon \rceil - 1)$.

Proof: Considering the definition of $f_{\mathsf{SO}}(R)$, we have

$$\mathcal{T}^*(R) = \mathcal{T}(f_{\mathsf{SO}}(R)) \leq \mathcal{T}(f_{\mathsf{MM}}(R)) \leq R \cdot \mathcal{M}(f_{\mathsf{MM}}(R)). \tag{13}$$

Leveraging inequality (8) in Lem. 1, it is

$$\begin{aligned} \mathcal{M}[f_{\mathsf{DF}}((1-\epsilon)R)] &\leq \frac{\mathcal{T}^*(R) - \mathcal{T}[f_{\mathsf{DF}}((1-\epsilon)R)]}{\epsilon R} \\ &\leq \frac{\mathcal{T}^*(R)}{\epsilon R} \overset{(a)}{\leq} \frac{R \cdot \mathcal{M}(f_{\mathsf{MM}}(R))}{\epsilon R} = \frac{\mathcal{M}(f_{\mathsf{MM}}(R))}{\epsilon}. \end{aligned} \tag{14}$$

Here inequality (a) follows (13).

Then we prove the *near-tightness* by claiming the bi-criteria maximum delay bound is $(1-\epsilon, \lceil 1/\epsilon \rceil - 1)$ in Fig. 2.

For any $\epsilon \in (0,1)$, it holds that $1/\epsilon > \lceil 1/\epsilon \rceil - 1$. Thus we can always find an $\alpha > 1$ such that $1/(\alpha\epsilon) = \lceil 1/\epsilon \rceil - 1$. We also denote $n = \lceil 1/\epsilon \rceil = 1/(\alpha\epsilon) + 1$. Clearly, we have $n \geq 2$. We construct a network as in Fig. 2, where there are n nodes and $2(n-1)$ links, and assume $R = 1$, $s = a_1$, and $t = a_n$.

All dashed links have a constant delay 1, while all solid links have a delay function as below

$$\mathcal{D}(x) = \begin{cases} 0 & \text{if } 0 \le x \le \frac{n-2}{n-1}, \\ [(x - \frac{n-2}{n-1})/(\frac{1-1/\alpha}{n-1})]^2 & \text{if } x > \frac{n-2}{n-1}. \end{cases} \quad (15)$$

It is straightforward to verify that $\mathcal{D}(x)$ satisfies our assumption that link delay functions are non-negative, non-decreasing, and differentiable. Moreover, we can obtain that $\mathcal{D}(\frac{n-2}{n-1}) = 0$, $\mathcal{D}(\frac{n-1-1/\alpha}{n-1}) = 1$, and $\mathcal{D}(R) = \mathcal{D}(1) > 1$.

We first show $\mathcal{M}(f_{\mathsf{MM}}(R)) = 1$ by directly constructing a feasible flow with maximum delay of 1. It is straightforward that $\mathcal{M}(f_{\mathsf{MM}}(R)) \ge 1$. We note that there are $(n-1)$ different $s-t$ paths containing exactly one dashed link. $f(R)$ routes $1/(n-1)$ rate on each of these $(n-1)$ paths. Hence, all solid links have a flow rate of $\frac{n-2}{n-1}$, thus with a delay $\mathcal{D}(\frac{n-2}{n-1}) = 0$. In $f(R)$, all flow-carrying paths have a path delay of 1 and its maximum delay is also 1. Therefore, $\mathcal{M}(f_{\mathsf{MM}}(R)) = 1$.

We then prove that $\mathcal{M}(f_{\mathsf{DF}}[(1-\epsilon)R]) = n-1 = \lceil 1/\epsilon \rceil - 1$ by constructing $f_{\mathsf{DF}}[(1-\epsilon)R]$ following Algorithm 1. According to Lem. 6, we get $f_{\mathsf{SO}}(R)$ by obtaining $f_{\mathsf{NE}}(R)$ with new link delay functions $\hat{\mathcal{D}}_e(x_e)$. For our example in Fig. 2, the new link delay function for dashed link is again the constant 1, but for solid link is $\hat{\mathcal{D}}(x) = \mathcal{D}(x) + x\mathcal{D}'(x)$ where $\mathcal{D}(x)$ is defined in Equation (15). It is easy to verify that $\hat{\mathcal{D}}(x)$ is non-decreasing and continuous over $x \ge 0$ and strictly increasing over $x \in [\frac{n-2}{n-1}, \frac{n-1-1/\alpha}{n-1}]$. Also we have

$$\hat{\mathcal{D}}(\frac{n-2}{n-1}) = 0, \quad \hat{\mathcal{D}}(\frac{n-1-1/\alpha}{n-1}) > 1.$$

Thus it is fair to define $\lambda \in (\frac{n-2}{n-1}, \frac{n-1-1/\alpha}{n-1})$ such that $\hat{\mathcal{D}}(\lambda) = 1$. By the definition of Nash equilibrium and Lem. 6, now we can claim that *in the system-optimal flow, all solid links have a rate of* λ *and all dashed links have a flow rate of* $(1-\lambda)$.

Then one possible path-based flow $f_{\mathsf{SO}}(R)$ is: a rate of $(1-\lambda)$ is assigned to the path p_1 containing all dashed links whose delay is $n-1$, and a rate of λ is assigned to the path p_2 containing all solid links whose delay is $(n-1)\mathcal{D}(\lambda) < n-1$.

Now following Algorithm 1, since p_1 is the slowest flow-carrying path and $1-\lambda > 1-\frac{n-1-1/\alpha}{n-1} = \epsilon = \epsilon R$, all ϵR rate will be deleted from path p_1. In addition, after deleting, path p_1 still contains a strictly positive rate, which implies

$$\mathcal{M}(f_{\mathsf{DF}}[(1-\epsilon)R]) = n-1. \quad (16)$$

Therefore, for this example, we have that

$$\frac{\mathcal{M}(f_{\mathsf{DF}}[(1-\epsilon)R])}{\mathcal{M}(f_{\mathsf{MM}}(R))} = n-1 = \lceil 1/\epsilon \rceil - 1,$$

which prove that the bound $(1-\epsilon, 1/\epsilon)$ is near tight. ■

Different from the system-optimal flow under a reduced rate, according to Thm. 3 and Thm. 4, $f_{\mathsf{DF}}[(1-\epsilon)R]$ is able to guarantee constant-ratio gaps for both maximum delay and average delay compared to the optimums, and can be obtained in polynomial time following our approach (Algorithm 1). Thus in applications, $f_{\mathsf{DF}}[(1-\epsilon)R]$ provides a solution where any user can experience a delay with known upper bound and guaranteed to be close to optimum, and in the mean time the total network will see an efficient utilization of resources since the average delay is also optimized.

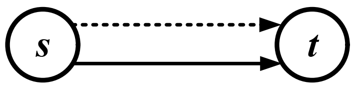


Fig. 3: A graph with two nodes and two parallel links. It is used to prove tightness in Thm. 5 and Thm. 7. The lower solid link has a delay function $D(x) = x^p$ for a large $p > 1$, and the upper dashed link has a constant delay 1.

C. *Comparing* $f_{\mathsf{NE}}[(1-\epsilon)R]$ *with Optimum*

Compared with SO, NE has an advantage of solving in polynomial time even for non-convex link delay functions (see Tab. I). In this subsection, for the Nash flow with a rate requirement $(1-\epsilon)R$, i.e., $f_{\mathsf{NE}}[(1-\epsilon)R]$, we prove both the average delay and the maximum delay must be bounded by a constant ratio of $1/\epsilon$ to the optimums in Thm. 5 and Thm. 6.

Theorem 5: Compared with $f_{\mathsf{SO}}(R)$, a bi-criteria average delay bound for the Nash flow $f_{\mathsf{NE}}[(1-\epsilon)R]$ is $(1-\epsilon, 1/\epsilon)$, which is tight.

Proof: The bound is a direct result from Lem. 5 which is the Theorem 3.2 in [14]. Tightness can be proved using Fig. 3. More details can be found in our technical report [15]. ■

Theorem 6: Compared with $f_{\mathsf{MM}}(R)$, a bi-criteria maximum delay bound for $f_{\mathsf{NE}}[(1-\epsilon)R]$ is $(1-\epsilon, 1/\epsilon)$.

The bound $(1-\epsilon, 1/\epsilon)$ is near tight in the sense that for any $\epsilon \in (0,1)$, there exists a problem instance such that the bi-criteria maximum delay bound is $(1-\epsilon, \lceil 1/\epsilon \rceil - 1)$.

Proof: By leveraging Lem. 4 and Thm. 5, we obtain

$$\begin{aligned}&\mathcal{M}(f_{\mathsf{NE}}[(1-\epsilon)R]) = \mathcal{A}(f_{\mathsf{NE}}[(1-\epsilon)R]) \\ &\le \frac{\mathcal{A}(f_{\mathsf{SO}}(R))}{\epsilon} \le \frac{\mathcal{A}(f_{\mathsf{MM}}(R))}{\epsilon} \le \frac{\mathcal{M}(f_{\mathsf{MM}}(R))}{\epsilon}. \quad (17)\end{aligned}$$

The near-tightness example is again Fig. 2 with same setting as Thm. 4. The intuition is that $f_{\mathsf{NE}}[(1-\epsilon)R]$ shall route $(1-\epsilon)R$ flow rate to the path with all solid links. ■

Compared with $f_{\mathsf{DF}}[(1-\epsilon)R]$, although $f_{\mathsf{NE}}[(1-\epsilon)R]$ can provide both maximum delay and average delay that are close to optimal, its average delay bound is worse. Thus, although the Nash flow is easy to obtain even for non-convex delay functions and provides an alternate routing to obtain both good maximum delay and good average delay, its average delay performance is outperformed by that of $f_{\mathsf{DF}}[(1-\epsilon)R]$ in theory.

D. *Comparing* $f_{\mathsf{MM}}[(1-\epsilon)R]$ *with Optimum*

Although MM is NP-hard, theoretically it still remains an interesting problem to evaluate its average delay performance as studied in [11], [12]. For the min-max flow with a rate requirement $(1-\epsilon)R$, i.e., $f_{\mathsf{MM}}[(1-\epsilon)R]$, in this subsection we show that its average delay is bounded by a constant ratio to the optimum in Thm. 7 and its maximum delay is no worse than the optimum with full rate requirement in Thm. 8.

Theorem 7: Compared with $f_{\mathsf{SO}}(R)$, a bi-criteria average delay bound for $f_{\mathsf{MM}}[(1-\epsilon)R]$ is $(1-\epsilon, 1/\epsilon)$, which is tight.

Proof: Leveraging Lem. 4 and Thm. 5, we have

$$\begin{aligned}\mathcal{A}(f_{\mathsf{MM}}[(1-\epsilon)R]) &\le \mathcal{M}(f_{\mathsf{MM}}[(1-\epsilon)R]) \le \mathcal{M}(f_{\mathsf{NE}}[(1-\epsilon)R]) \\ &= \mathcal{A}(f_{\mathsf{NE}}[(1-\epsilon)R]) \le \mathcal{A}(f_{\mathsf{SO}}(R))/\epsilon. \quad (18)\end{aligned}$$

The tightness of the delay bound can be proven using Fig. 3. More details are presented in our report [15]. ∎

Theorem 8: Compared with $f_{\mathsf{MM}}(R)$, a bi-criteria maximum delay bound for $f_{\mathsf{MM}}[(1-\epsilon)R]$ is $(1-\epsilon, 1)$, which is tight.

Proof: The bound can be proved easily following Lem. 2 and its tightness is a direct result from Fig. 1. ∎

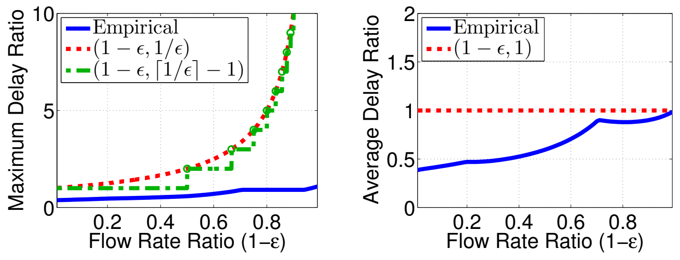
We already know different path-based system-optimal flows under the same rate requirement may have different maximum delay, and in the worst case the maximum delay is upper bounded just by a delay-function-dependent ratio (Thm. 2) if we solve SO directly with a reduced rate. Similarly, different path-based min-max flows under the same rate requirement could have different average delay, too. However, according to Thm. 7 and Thm. 8, both the maximum delay result and the average delay result by solving MM with a reduced rate are constant-bounded to the two optimums.

Overall, recall that according to existed studies [11], [12], a pessimistic result says that a flow cannot simultaneously obtain maximum delay and average delay even within constant-ratio gaps to the optimums for general network topology and arbitrary link delay function. However, as analyzed in this section, our results pose an optimistic note that maximum delay and average delay are in fact "largely" compatible, because there exist many different flows, i.e. $f_{\mathsf{DF}}[(1-\epsilon)R]$, $f_{\mathsf{NE}}[(1-\epsilon)R]$ and $f_{\mathsf{MM}}[(1-\epsilon)R]$, that can theoretically guarantee both maximum delay and average delay to be close to optimums even in the worst case, only in the cost of losing ϵ fraction of flow rate, calling for future studies on improving the network delay performance in more practical delay-sensitive applications by sacrificing a controllable portion of the flow rate.

V. Experiments

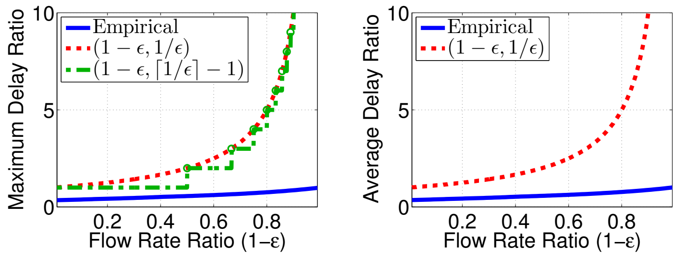
In this section we investigate the maximum delay and average delay for the solutions of SO, NE and MM with sacrificed flow rate, compared to the optimal maximum delay and average delay with full rate. We use a real-world continent-scale network *GEANT*, the main European research and education computer network with 45 nodes and 57 undirected links [18]. The capacity c_e (Gbps) for a link $e \in E$ is set according to the GEANT map [18], and $c_e \in \{5, 10, 20, 30, 100\}$. Each undirected link is treated as two directed links that operate independently and have identical capacities, same to the setting in [9]. Due to the page limit, the detailed network topology with link capacity is shown in our technical report [15]. We assume that there are 1000 video conferencing sessions from the top-left node Iceland (source) to the bottom-right node Israel (receiver) in GEANT. We further assume the link delay function to be the queuing delay formulated as (1) which is also used for Nash equilibrium studies in [9].

Our test environment is an Intel Core i5 (2.40 GHz) processor with 8 GB memory running Windows 64-bit operating system. The edge-based system-optimal flow and Nash flow are modeled as convex programs and solved using *CVX* in *Matlab*. All the other experiments including flow decomposition and Algorithm 1 are implemented in C++. Note that we cannot find the exact min-max flow $f_{\mathsf{MM}}(R)$ since it is NP-hard and the network is a dense graph. Instead we find a



(a) Maximum delay (b) Average delay

Fig. 4: Empirical and theoretical bi-criteria delay bounds for $f_{\mathsf{DF}}[(1-\epsilon)R]$.



(a) Maximum delay (b) Average delay

Fig. 5: Empirical and theoretical bi-criteria delay bounds for $f_{\mathsf{NE}}[(1-\epsilon)R]$.

feasible flow $\hat{f}_{\mathsf{MM}}(R)$ which is a min-max flow under rate R in a sub-network of GEANT to approximate $f_{\mathsf{MM}}(R)$. Comparing $\hat{f}_{\mathsf{MM}}(R)$ to $f_{\mathsf{MM}}(R)$ from maximum delay, the average (resp. maximum, minimum) relative error in our experiments is bounded above by 1.73% (resp. 4.58%, 0.21%) for all R from 0 to 10 with a step of 0.1. The relative error is defined as:

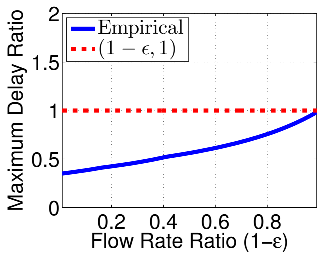
$$\text{Relative Error} = \frac{\mathcal{M}(\hat{f}_{\mathsf{MM}}(R)) - \mathcal{M}(f_{\mathsf{MM}}(R))}{\mathcal{M}(f_{\mathsf{MM}}(R))}.$$

Hence, for GEANT $\hat{f}_{\mathsf{MM}}(R)$ is a very good approximation to $f_{\mathsf{MM}}(R)$. Details of finding $\hat{f}_{\mathsf{MM}}(R)$ and the correctness of the upper bounded relative error are presented in our report [15].

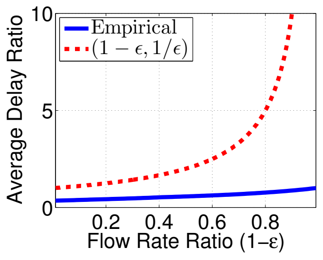
A. Maximum Delay Ratios compared with the Optimum

We assume that the video conferencing traffic demand for each session is 10Mbps, leading to a total rate requirement $R = 10$Gbps from the source to the receiver. As a benchmark, without sacrificing flow rate, we first obtain the maximum delay ratio comparing $f_{\mathsf{SO}}(R)$ to $\hat{f}_{\mathsf{MM}}(R)$, which is 1.128, and the maximum delay ratio comparing $f_{\mathsf{NE}}(R)$ to $\hat{f}_{\mathsf{MM}}(R)$, which is 1.017. In this section, we show that we can obtain a better maximum delay bound when we can sacrifice a small portion (3% for example) of rate requirement. We remark that 3% loss rate is very acceptable for video conferencing engines with loss protection/recovery and error resilience capabilities [19].

In our experiment, we increase the sacrificing ratio ϵ from 0.01 to 0.99 with a step of 0.01. Fig. 4a presents the maximum delay ratio comparing $f_{\mathsf{DF}}[(1-\epsilon)R]$ to $\hat{f}_{\mathsf{MM}}(R)$, which plots the theoretical near-tight gap $(1-\epsilon, 1/\epsilon)$ (red dotted line), the theoretical lower bound $(1-\epsilon, \lceil 1/\epsilon \rceil - 1)$ for the tight



(a) Maximum delay



(b) Average delay

Fig. 6: Empirical and theoretical bi-criteria delay bounds for $\hat{f}_{\mathsf{MM}}[(1-\epsilon)R]$.

gap (green dashed line), and the empirical maximum delay ratio (blue solid line). Clearly the empirical maximum delay bound is substantially smaller than the proposed theoretical maximum delay bound, implying that empirical maximum delay of $f_{\mathsf{DF}}[(1-\epsilon)R]$ with reduced rate is close to or even smaller than the maximum delay of $\hat{f}_{\mathsf{MM}}(R)$. Besides, we can observe that more sacrifice of flow rate results in better maximum delay. In addition, for the acceptable small loss in flow rate for video conferencing, i.e., $\epsilon = 0.03$, the maximum delay ratio is 1.007, which is much better than that of the system-optimal flow without rate loss, i.e., 1.128.

Fig. 5a shows the results for the maximum delay from $f_{\mathsf{NE}}[(1-\epsilon)R]$. For a small loss of rate ($\epsilon = 0.03$), the maximum delay ratio is 0.953 which is better than 1.017 when we do not sacrifice rate requirement. The ratio 0.953 is below 1 because $f_{\mathsf{NE}}[(1-\epsilon)R]$ is the flow with smaller rate and thus we could obtain a maximum delay strictly better than the optimum for flows with larger full rate. We observer similar results from $\hat{f}_{\mathsf{MM}}[(1-\epsilon)R]$ (Fig. 6a).

B. Average Delay Ratios compared with the Optimum

Similar to Sec. V-A, we set $R = 10$Gbps and ϵ is increased from 0.01 to 0.99 with a step of 0.01.

Fig. 4b illustrates the average delay ratio comparing $f_{\mathsf{DF}}[(1-\epsilon)R]$ to $f_{\mathsf{SO}}(R)$. Obviously, the average delay of $f_{\mathsf{DF}}[(1-\epsilon)R]$ is always no greater than that of $f_{\mathsf{SO}}(R)$. Interestingly the average delay of $f_{\mathsf{DF}}[(1-\epsilon)R]$ is not monotonic increasing w.r.t. the flow rate ratio $(1-\epsilon)$. From Fig. 5b (resp. Fig. 6b), we can see that $f_{\mathsf{NE}}[(1-\epsilon)R]$ (resp. $\hat{f}_{\mathsf{MM}}[(1-\epsilon)R]$) provides a much better empirical average delay gap than the theoretical bound $(1-\epsilon, 1/\epsilon)$.

Moreover, even for a small rate loss, for example $\epsilon = 0.03$, the average delay ratio is 0.972 both for $f_{\mathsf{NE}}[(1-\epsilon)R]$ and $\hat{f}_{\mathsf{MM}}[(1-\epsilon)R]$. This again shows the benefit of controllably sacrificing flow rate: without sacrificing rate, the experimental average delay ratio is 1.037 for both $f_{\mathsf{NE}}(R)$ and $\hat{f}_{\mathsf{MM}}(R)$.

Overall from Fig. 4, Fig. 5 and Fig. 6, we can see that for the three network delay optimization problems, observed empirical delay gaps are mush smaller than our bi-criteria bounds (see Tab. III) which are guaranteed even in the worst theoretical case, verifying the correctness of our proposed theoretical results and further calling for possible future studies to improve our delay gaps with more practical parameters involved for network delay optimization applications with a tolerance of a small fraction of traffic loss.

VI. Conclusions

We consider the scenario where a source streams a flow at fixed rate to a receiver across a network, and transmission over a link incurs a delay modeled as a non-negative, non-decreasing and differentiable function of the link aggregated transmission rate. We study three fundamental network delay optimization problems, concerning two popular metrics, namely maximum delay and average delay experienced by the flow. The three problems are (i) minimizing the maximum delay, denoted as problem MM, (ii) minimizing the average delay, denoted as problem SO, and (iii) finding the Nash equilibrium, denoted as problem NE. Our focus is on understanding the fundamentals of the two delay metrics.

For general network and arbitrary link delay function, a well-known pessimistic result says that a flow cannot simultaneously achieve optimal maximum delay and average delay, or even within constant-ratio gaps to the optimums. But we design three solutions, all of which can deliver $(1-\epsilon)$ fraction of the flow with maximum delay and average delay simultaneously within $1/\epsilon$ to the optimums. Hence, our results pose an optimistic note on the fundamental compatibility of the two delay metrics. The ratio $1/\epsilon$ is independent to the network size and link delay function, and is at least near-tight. As byproducts, we derive two polynomial-time bi-criteria approximation algorithms to solve the NP-hard problem MM.

Simulations based on real-world continent-scale network topology verify our theoretical findings. The empirical delay gaps observed under practical settings are much smaller than $1/\epsilon$, which is the proposed theoretical delay gap guaranteed even under the worst case setting upon sacrificing ϵ fraction of the flow rate, calling for future studies to improve our proposed gaps with more practical parameters involved in network delay optimization applications. Our results are of particular interests to delay-centric networking applications that can tolerate a small fraction of traffic loss, including cloud video conferencing that recently attracts substantial attention.

References

[1] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, “Data center tcp (dctcp),” in *ACM SIGCOMM*, 2010.

[2] B. Vamanan, J. Hasan, and T. Vijaykumar, “Deadline-aware datacenter tcp (d2tcp),” in *ACM SIGCOMM*, 2012.

[3] Y. Liu, D. Niu, and B. Li, “Delay-optimized video traffic routing in software-defined interdatacenter networks,” *IEEE Transactions on Multimedia*, 2016.

[4] M. H. Hajiesmaili, L. T. Mak, Z. Wang, C. Wu, M. Chen, and A. Khonsari, “Cost-effective low-delay design for multi-party cloud video conferencing,” *IEEE Transactions on Multimedia*, 2017.

[5] WebEx. [Online]. Available: https://blog.webex.com/2016/01/five-reasons-to-join-a-webex-now/

[6] Skype. [Online]. Available: https://news.microsoft.com/bythenumbers/skype-calls

[7] Cisco. [Online]. Available: http://www.cisco.com/c/dam/en/us/solutions/collateral/collaboration/midsize-collaboration-solutions/high-growth-innovative-companies.pdf

[8] ITU, “Series g: Transmission systems and media, digital systems and networks,” 2003.

[9] B. Grimmer and S. Kapoor, “Nash equilibrium and the price of anarchy in priority based network routing,” in *IEEE INFOCOM*, 2016.

[10] L. Kleinrock, "Analytic and simulation methods in computer network design," in *Proceedings of the spring joint computer conference*, 1970.
[11] J. R. Correa, A. S. Schulz, and N. E. Stier-Moses, "Fast, fair, and efficient flows in networks," *Operations Research*, 2007.
[12] J. R. Correa, A. S. Schulz, and N. E. S. Moses, "Computational complexity, fairness, and the price of anarchy of the maximum latency problem," in *IPCO*, 2004.
[13] T. Roughgarden, "The maximum latency of selfish routing," in *SODA*, 2004.
[14] T. Roughgarden and É. Tardos, "How bad is selfish routing?" *Journal of the ACM*, 2002.
[15] "A tale of two metrics in network delay optimization," *Technical Report*, 2017. [Online]. Available: https://www.dropbox.com/s/fgp3p3kaz650hhb/paper-TR.pdf?dl=0
[16] M. Beckmann, C. B. McGuire, and C. B. Winsten, "Studies in the economics of transportation," Tech. Rep., 1956.
[17] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian Journal of Mathematics*, 1956.
[18] "The pan-european research and education network," 2017. [Online]. Available: https://geant3plus.archive.geant.net/Resources/Media_Library/Documents/GEANT_Project_TopologyMAR14_Web.pdf
[19] I. M. Weinstein, "Polycoms lost packet recovery (lpr) capability," *Wainhouse Research*, 2008.

Appendix

A. Straightforward or Existing Useful Lemmas

First, due to the non-decreasing property for link delay functions, directly it holds that the optimal maximum delay and optimal total delay are non-decreasing w.r.t. the rate.

Lemma 2: $\mathcal{M}^*(R)$ is non-decreasing w.r.t. the rate R.

Lemma 3: $\mathcal{T}^*(R)$ is non-decreasing w.r.t. the rate R.

Second, for Nash flows we have the following two lemmas.

Lemma 4: The maximum delay and average delay of a Nash flow are equal, i.e.,

$$\mathcal{M}[f_{\mathsf{NE}}(R)] = \mathcal{A}[f_{\mathsf{NE}}(R)] = \mathcal{T}[f_{\mathsf{NE}}(R)]/R. \tag{19}$$

Proof: By the definition of a Nash flow (Definition 1), all flow-carrying paths share the same path delay. ■

Lemma 5: [14, Theorem 3.2] If f' is any flow with $|f'| = (1+\beta)R$ where $\beta > 0$, then the total delay of Nash flow $f_{\mathsf{NE}}(R)$ is upper bounded by

$$\mathcal{T}[f_{\mathsf{NE}}(R)] \le \frac{1}{\beta}\mathcal{T}(f').$$

In addition, the bound is tight.

Lem. 5 presents a tight total delay bound between the Nash flow $f_{\mathsf{NE}}(R)$ with rate requirement R and any feasible flow with a larger rate requirement $(1+\beta)R$ where $\beta > 0$.

Third, we can find the system-optimal flow by obtaining a Nash flow or vice verse, according to the following lemma.

Lemma 6: [14, Corollary 2.5] If for any link e, function $x_e \cdot \mathcal{D}_e(x_e)$ is convex, then a flow is a system-optimal flow in the original graph with link delay functions $\mathcal{D}_e(x_e)$ if and only if it is a Nash flow in the same graph with link delay functions $\hat{\mathcal{D}}_e(x_e) = \mathcal{D}_e(x_e) + x_e\mathcal{D}'_e(x_e)$.

B. Proof of Our Proposed Lemma 1

Note in this subsection $f_{\mathsf{SO}}(R)$ denotes the *specific path-based system-optimal flow returned* in line 4 of Algorithm 1.

According to Algorithm 1, $f_{\mathsf{DF}}[(1-\epsilon)R]$ is obtained by iteratively deleting ϵR rate from $f_{\mathsf{SO}}(R)$. Suppose that there are in total K iterations to get $f_{\mathsf{DF}}[(1-\epsilon)R]$. We use f_k to represent the flow at the beginning of the k-th iteration (or equivalently, at the end of the $(k-1)$-th iteration) for $1 \le k \le K+1$. Obviously, $f_1 = f_{\mathsf{SO}}(R)$, $f_{K+1} = f_{\mathsf{DF}}[(1-\epsilon)R]$.

We denote P_k as the set of of flow-carrying paths in flow f_k, and $p_k \in P_k$ as the slowest flow-carrying path in f_k. According to Algorithm 1, in the k-th iteration, we delete some rate, say $x_k > 0$, from p_k.

Since all link delay functions are non-decreasing, the path delay of all flow-carrying paths cannot increase with reduced flow rate. Thus, the maximum delay cannot increase, i.e.,

$$\mathcal{M}(f_{k+1}) \le \mathcal{M}(f_k), \tag{20}$$

implying that

$$\mathcal{M}(f_{\mathsf{DF}}[(1-\epsilon)R]) \le \mathcal{M}(f_{\mathsf{SO}}(R)). \tag{21}$$

Considering the total delay, for any k, we have

$$\begin{aligned}
&\mathcal{T}(f_k) = \sum_{e\notin p_k}[x_e\mathcal{D}_e(x_e)] + \sum_{e\in p_k}[x_e\mathcal{D}_e(x_e)] \\
&= \sum_{e\notin p_k}[x_e\mathcal{D}_e(x_e)] + \sum_{e\in p_k}[(x_e - x_k)\mathcal{D}_e(x_e)] + x_k\sum_{e\in p_k}\mathcal{D}_e(x_e) \\
&\overset{(a)}{=} \sum_{e\notin p_k}[x_e\mathcal{D}_e(x_e)] + \sum_{e\in p_k}[(x_e - x_k)\mathcal{D}_e(x_e)] + x_k\mathcal{M}(f_k) \\
&\overset{(b)}{\ge} (\sum_{e\notin p_k}[x_e\mathcal{D}_e(x_e)] + \sum_{e\in p_k}[(x_e - x_k)\mathcal{D}_e(x_e - x_k)]) + x_k\mathcal{M}(f_k) \\
&\overset{(c)}{=} \mathcal{T}(f_{k+1}) + x_k\mathcal{M}(f_k) \overset{(d)}{\ge} \mathcal{T}(f_{k+1}) + x_k\mathcal{M}[f_{\mathsf{DF}}((1-\epsilon)R)].
\end{aligned} \tag{22}$$

In (22), equality (a) holds because $\sum_{e\in p_k}\mathcal{D}_e(x_e)$ is the path delay of the slowest flow-carrying path p_k. Inequality (b) follows the non-decreasing property of link delay functions. Equality (c) holds because flow f_{k+1} is the resulting flow when flow f_k deletes x_k rate in path p_k. Inequality (d) comes from (20) and $f_{K+1} = f_{\mathsf{DF}}[(1-\epsilon)R]$. We then do summation for (22) over $k \in [1, K]$, and get

$$\begin{aligned}
\mathcal{T}^*(R) &= \mathcal{T}[f_{\mathsf{SO}}(R)] = \mathcal{T}(f_1) \\
&\ge \mathcal{T}(f_{K+1}) + \left(\sum_{k=1}^{K} x_k\right)\cdot\mathcal{M}[f_{\mathsf{DF}}((1-\epsilon)R)] \\
&= \mathcal{T}[f_{\mathsf{DF}}((1-\epsilon)R)] + \epsilon R\cdot\mathcal{M}[f_{\mathsf{DF}}((1-\epsilon)R)],
\end{aligned}$$

which proves (8).

For the average delay, we have

$$\begin{aligned}
\mathcal{A}(f_k) &= \frac{\mathcal{T}(f_k)}{|f_k|} \ge \frac{\mathcal{T}(f_{k+1}) + x_k\mathcal{M}(f_k)}{|f_k|} \quad \texttt{//by}\ (22) \\
&= \frac{\mathcal{A}(f_{k+1})(|f_k| - x_k) + x_k\mathcal{M}(f_k)}{|f_k|} \\
&= \frac{\mathcal{A}(f_{k+1})|f_k| + x_k(\mathcal{M}(f_k) - \mathcal{A}(f_{k+1}))}{|f_k|} \\
&\ge \frac{\mathcal{A}(f_{k+1})|f_k| + x_k(\mathcal{M}(f_{k+1}) - \mathcal{A}(f_{k+1}))}{|f_k|} \quad \texttt{//by}\ (20) \\
&\ge \frac{\mathcal{A}(f_{k+1})|f_k|}{|f_k|} = \mathcal{A}(f_{k+1}),
\end{aligned} \tag{23}$$

which proves (7) by iterating k from 1 to K.